OPIC
OFFICE DE LA PROPRIÉTÉ INTELLECTUELLE DU CANADA

CIPO
CANADIAN INTELLECTUAL PROPERTY OFFICE

(12) (19) (CA) **Demande-Application**

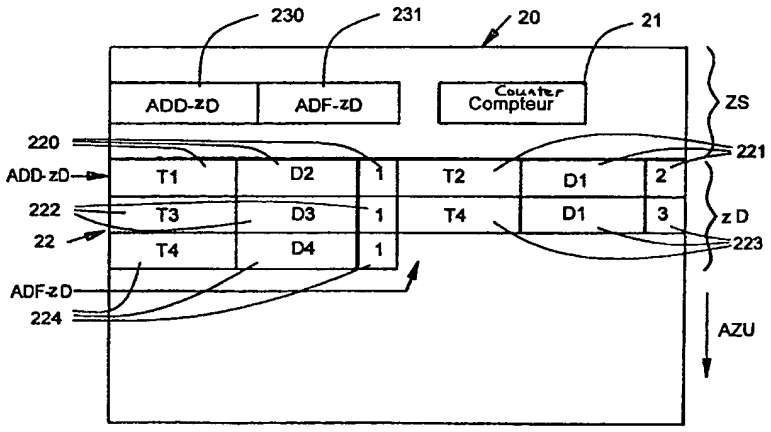
(21) (A1) **2,247,474**
(86) 1997/12/23
(87) 1998/07/02

(72) LAPIE, Ronan, FR
(71) BULL CP8, FR

(51) Int.Cl.[6] G06F 11/22
(30) 1996/12/24 (96/15997) FR

(54) **TERMINAL ET PROCEDE D'AUTODIAGNOSTIC OU DE SUPERVISION ET OBJET PORTATIF UTILISE DANS UN TEL TERMINAL OU PROCEDE**

(54) **SELF-DIAGNOSTIC OR SUPERVISORY TERMINAL AND METHOD AND PORTABLE OBJECT USED IN SUCH TERMINAL OR METHOD**



(57) Terminal doté d'un programme d'application, d'au moins une sortie constitué soit par un affichage, soit par une imprimante, soit par un réseau de communication, soit par un objet portable et coopérant avec un objet portable doté d'un zone de mémoire non-volatile (ZD) contenant des données et comportant un lecteur communiquant avec ledit objet portable, caractérisé en ce que l'appareil comporte des moyens de lecture ou de stockage, dans sa mémoire, de données (Ti, Dj, Sk) d'autodiagnostic ou de supervision et un moyen d'émission desdites données vers des sorties (1-4) spécifiées en fonction d'informations fournies par les données d'autodiagnostic ou de supervision suite à l'exécution d'au moins une tâche Tt de son programme d'application en relation avec l'objet portable.

(57) The invention concerns a terminal equipped with an application programme, with at least one output consisting of either a display or a printer, or a communication network, or a portable object and co-operating with a portable object equipped with a non-volatile memory zone (ZD) containing data and comprising a reader communicating with said portable object. The invention is characterised in that the apparatus comprises means for reading or storing in its memory self-diagnostic or supervisory data (Ti, Dj, Sk) and means for transmitting said data to specific outputs (1-4) on the basis of information supplied by the self-diagnostic or supervisory data following the execution of at least one task (Tt) of its application programme relative to the portable object.

CA 02267474 1998-08-20

ABSTRACT

Inventor: Ronan LAPIE

Terminal equipped with an application program, with at least one output constituted either by a display, or by a printer, or by a communications network, or by a portable object, and cooperating with a portable object equipped with a non-volatile memory area (ZD) containing data, and comprising a reader which communicates with said portable object, characterized in that the device comprises means for reading or storing, in its memory, self-diagnostic or supervisory data (Ti, Dj, Sk) and means for sending said data to outputs (1-4) specified as a function of information supplied by the self-diagnostic or supervisory data following the execution of at least one task Tt of its application program in connection with the portable object.

CA 02247474 1998-08-20

TERMINAL AND PROCESS FOR SELF-DIAGNOSIS OR SUPERVISION AND PORTABLE OBJECT USED IN A TERMINAL OR PROCESS OF THIS TYPE

The present invention relates to a terminal and a process for self-diagnosis or supervision and the portable object of the microcircuit card type used in a process or terminal or reader of this type. These objects comprise a central processor, a program memory containing an executable code constituting the operating system, a non-volatile and programmable data memory, and one or more communication interfaces. The terminals are devices equipped with an interface compatible with those of the objects; they have a central processor and software that is capable of communicating and operating on the data issuing from the non-volatile memory of the portable object.

In general, the terminals are equipped with specific software corresponding to their utilization, for example portable payment terminals are equipped with an operating program of the banking type. This software is produced or specified by the institution that manages this application; in the example cited, this is a banking institution. This institution is generally not the manufacturer of the terminal; it purchases or orders the manufacture of the hardware part, that is the terminal, and it installs the specific program into it in order to configure its terminal for its own application. The banking institution thereby has the advantage of purchasing a standard, and therefore inexpensive, product and adapting it according to its needs. The manufacturer offers a basic model which can be suited to a plurality of applications, which enables it to expand its market.

The institution running a given application may wish to use a plurality of card reader terminal models; it is not desirable to develop application software for each of the terminals. This has led manufacturers to implement a basic software layer that ensures the interface between the hardware and the application software. This software layer enables the same application software to be adapted to different terminals. One way to do this is to create an interpreter so that the institution can develop

its application in a well-known high-level language, nearly independent of the constraints of the hardware. Another way to do this is to set up a low-level software layer that manages all the hardware input-outputs and to make available to the operating institution a library of primitives that the application software will call.

In all cases, it must be possible to validate or to test the terminal in its entirety. The validation or testing of the terminal must take into account both parts: the hardware with its basic software and the application software. A self-test makes it possible to verify each device of the terminal; it is generally constituted by a routine implemented in the basic software. The testing of the application software must be done in the laboratory; it is important in this type of application to properly verify the operation of the program before it is placed in service. However, the multiplicity of cards results in a very large number of specific cases that are not reproducible in the laboratory. The object of the following invention is to validate or test, under normal usage conditions, the operation of application software.

To this end, the invention relates to a terminal equipped with an application program, with at least one output constituted either by a display, or by a printer, or by a communication network, or by a portable object, and cooperating with a portable object equipped with a non-volatile memory area containing data, and comprising a reader which communicates with this portable object, characterized in that the device comprises means for reading or storing in its memory diagnostic or supervisory data and means for sending these data to outputs specified as a function of information supplied by the self-diagnostic or supervisory data following the execution of at least one task of its application program in connection with the portable object.

According to another characteristic, the means for sending the self-diagnostic data are activated a certain number of times.

According to another characteristic, the means for reading

the self-diagnostic or supervisory data comprise means for writing in the portable object connected to the device.

According to another characteristic, the self-diagnostic or supervisory data are constituted by at least one triplet of information corresponding, for a first piece of information, to a predetermined task of the application program, for the second piece of information to a data type correlated to the task executed and to be presented to an output, and for the third piece of information to a value for specifying the output to which the data type must be presented among those present in the terminal.

According to another characteristic, the device has a means for testing for the presence of self-diagnostic or supervisory data in a portable object and for initiating the reading and the storage of these data in a specific area ZTD of the memory of the terminal.

According to another characteristic, the terminal comprises means for entering self-diagnostic or supervisory data into a portable object.

Another object of the invention is to propose a process for supervising the operation of a terminal or for the self-diagnosis of a terminal.

This object is achieved due to the fact that the self-diagnostic or supervisory process, from at least one triplet of information corresponding, for a first piece of information, to a predetermined task of an application program executed either by a portable object or by a terminal, for the second piece of information to a data type correlated to the task executed and to be presented to an output, and for the third piece of information to a value for specifying the output among those present in the terminal, characterized in that it is comprised of:

- executing a task of the application program in the terminal;

- testing an indicator either in the terminal or in the portable object to determine whether a self-diagnostic or

supervisory function is operational, then in the case of a positive response;

- searching in the memory of either the portable object or the terminal to see if among the triplets of information stored there is a triplet wherein the first piece of information corresponds to the predetermined task executed by the terminal or the card;

- sending to the output specified by the triplet thus read the value of the datum correlated to the task executed and to be labelled by the second piece of information in the triplet.

According to another characteristic, the process comprises a testing step comprised of determining whether there are other tasks to be executed, and following the execution of these tasks, searching for all of the triplets of information corresponding to the execution of this task.

According to another characteristic, the process comprises a step for reading from a portable object storing in its non-volatile memory a plurality of triplets and a step for storing these triplets in a non-volatile memory area of the terminal, followed by a step for activating an indicator of an active self-diagnostic or supervisory function.

According to another characteristic, the process comprises a testing step for determining whether the portable object is a card specific to the self-diagnostic or supervisory function or a so-called general-purpose card.

According to another characteristic the self-diagnostic or supervisory data are constituted by a fourth field of information containing in the portable object initially the write address (Adr-V), the number of octets to be written (Nb-V), and after the self-diagnosis operation, the value to be written (Val).

Another object of the invention is to propose a portable object that can be used with the terminal and the self-diagnostic or supervisory process.

This object is achieved due to the fact that the portable object is a microprocessor card operating by means of an

operating system stored in the card and comprising a non-volatile memory containing at least one triplet of information in a predetermined area of this non-volatile memory whose location is defined by address fields located in the memory part used to store the operating system.

According to another characteristic, the part of non-volatile memory used to store the operating system also comprises, in a memory field, a piece of information constituting a counter of utilizations of the self-diagnostic function.

According to another characteristic, the memory area storing the operating system comprises a field which makes it possible to store an indicator of the activation of the self-diagnostic or supervisory function.

Other characteristic and advantages of the present invention will become more clearly apparent with the reading of the following description given in reference to the appended drawings, in which:

Fig. 1 is an explanatory table of the triplets constituted by elementary tasks, data types and output types that can be associated with the execution of each of the tasks of an application program;

Fig. 2 is a diagram of the non-volatile memory areas used in a portable testing object necessary to the implementation of the process of the invention;

Fig. 3 shows the various steps in the execution of the programs for initializing the terminal and executing the self-diagnostic function;

Fig. 4 shows schematic view of the areas for storing information in the non-volatile memory of a so-called general-purpose portable object;

Fig. 5 shows the various steps in the execution of a program for initializing the terminal and executing the self-diagnostic function in this terminal with a so-called general-purpose card.

One way to implement the invention is comprised, in a first variant, of using a microprocessor card operating by means of an

operating system stored in the card and constituting a "so-called smart card" that is pre-initialized with the self-diagnostic data, and of having these data taken into account by the terminal when intending to test the application software.

An application software in a terminal can be broken down into elementary tasks which occur at predetermined moments. For example, for a banking application, a transaction can be broken down into the following elementary tasks (Ti): Verification (T1, Fig. 1) of whether the card inserted is authorized, Authentication (T2) of the bearer, Acquisition (T3) of the data of the transaction in the terminal, Writing of these data into the microcircuit of the card, Validation (T4) of the transaction in the terminal and the card.

In addition, and always during a transaction, the application software manipulates data; these data can be used temporarily, like the Code created by the bearer (Cp), which is stored in the memory of the terminal, or the identity of the bearer (Ip), which is stored in the card, or like the Amount of the transaction (Mt) or the Date of the transaction (Dt), which are stored in the terminal and the card. During the execution of each elementary task, each of these data may be initialized, modified or unchanged. The self-diagnostic utility function of the application software is comprised of verifying the data of the transaction at the time of certain tasks, under normal usage conditions. This function can be performed either by the basic software or by the application software.

To do this, the operator responsible for the test develops a grid composed, on one side, of the identifiable elementary tasks, designated Ti, and on the other side of data Dj constituted by information such as: Cp, Ip, Mt and Dt. Fig. 1 shows an example of such a grid. In order to test the correct execution of a transaction, the operator chooses to verify the value of certain data during the execution of specific tasks. This involves associating a datum Dj with a task Ti; these associations are symbolized by crosses in the grid of Fig. 1. A third piece of

information Sk is added. The value of this code indicates the type of output used to which the datum to be verified is to be sent: to the network when Sk is at a first value, for example "1" (Sk = 1), to the printer when Sk is at a second value, for example "2" (Sk = 2), or to the screen when Sk is at a third value, for example "3" (Sk = 3). The operator enters the triplets of information (Ti, Dj, Sk) into a special so-called diagnostic central processor, which central processor is equipped with a card reader. The diagnostic software is configured in accordance with the application to be tested so that the triplets (Ti, Dj, Sk), when captured, are identified on the screen by the precise indication of the elementary tasks and the data to be verified, and not the numeric labels Ti, Dj, Sk.

The card containing the self-diagnostic data is either a special card or a general-purpose card normally used for an application. A detailed description of an embodiment is given for each case.

The first case described is that in which a special card called a "test card" (20, Fig. 2) is used to contain the self-diagnostic data. A security procedure is implemented to prevent a defrauder from being able to use a card of this type in an unauthorized manner. The test card contains in the secret memory area, not shown, a secret so-called diagnostic code "KD." This secret code must first be presented to the card, which verifies it and, if it is equal to a reference code, authorizes the writing of the self-diagnostic data into the programmable memory of the card.

While providing for the storage of the self-diagnostic data, the non-volatile programmable memory of the test card also has, in addition to the system area ZS which contains the operating system of the card and the other usable area (AZU) which allows other types of storage, an area (22) called "ZD." It is in this area that the triplets (Ti, Dj, Sk) are stored in succession. Thus, a first area (220) of a memory allows the storage of the first triplet T1, D2, 1; a second area (221) allows the storage

of the second triplet T2, D1, 2; a third area (222) allows the storage of the third triplet T3, D3, 1; a fourth area (223) allows the storage of the fourth triplet D4, D3, 3; a fifth area (224) allows the storage of the triplet T4, D4, 1; T1, T2, T3, T4, D1, D2, D3, D4 respectively representing the information in Fig. 1. It is obvious that the portable object can comprise more or fewer triplets depending on the type of supervision or self-diagnosis that it is desirable to perform on the tasks executed by the application program. The area ZD is labelled by its start address "ADD_ZD" and its end address "ADF-ZD"; the two address values are stored in the part (230, 231) of the programmable memory allocated to the operating system.

The non-volatile programmable memory is of the EPROM, EEPROM, FeRAM, SRAM or FLASH type. Fig. 2 describes the organization of this memory using the information cited in Fig. 1. Advantageously, the datum Dj is the physical address of the datum to be verified in the working memory of the terminal.

Once programmed, the test card is inserted into the terminal in which the self-diagnostic function must be run. Fig. 3 is a flow chart illustrating the chronology of the events of the program, constituted by a wait and test sequence (1, 2, 3), the test triggering, as a function of the result, either a sequence for loading the terminal with the self-diagnostic data (4 through 7, Fig. 3), or a sequence for executing the self-diagnostic program (8 through 16, Fig. 3), which is incorporated either into the basic software of the terminal or into the application software. Step 1 is the initialization of the terminal after it is powered up, and step 2 is the phase for waiting for an order or an insertion of a card. In step 3, the terminal tests whether the card inserted into the reader is a general-purpose card, and in step 4 whether the card is a test card. In the latter case, the terminal executes in step 5 a procedure for authenticating the card by means of a reference code or by means of a standard challenge-response authentication scheme using an algorithm and a secret key (KD).

Once the test card has been identified and authenticated, the program of the terminal reads in step 6 the information contained in the area ZD. The selection and the location of the triplets are performed with the aid of the two pointers ADD_ZD and ADF-ZD. The triplets (Ti, Dj, Sk) read successively in the area ZD are stored in the same order in an area of the memory of the terminal called ZTD. Once the last triplet has been stored in the area ZTD, the terminal program, in step 7, sets a self-diagnostic indicator "Ind_DT" in the memory of the terminal to the active state. Then the terminal program loops back to wait for a command or another insertion of a card in step 2.

A new card is inserted; it is a general-purpose card compatible with the application run by the terminal. As stated above, the application software in the terminal is broken down into elementary tasks Tt which can be executed individually (step 8). At the end of the execution of each task, which can be labelled by the code Tt, the application program, in step 9, tests the indicator Ind_DT of the terminal. If it is inactive, the self-diagnostic function is not operational, and the program continues to execute the other tasks. If the indicator Ind_DT is active, the program of the terminal, in step 10, searches the area ZTD of the memory of the terminal for the first triplet (Ti, Dj, Sk) for which Tt = Ti, that is, to see if there is a datum to be tested as a result of the task that has just been executed. If yes, in step 11, the value "Val" of this datum (Dj) is temporarily stored in the memory of the terminal, and as a function of the value of Sk, it is processed in the following way:

If Sk is equal to "1" (step 12), the datum Dj must be sent to the network. In this case, a block of three data is constituted by the value of the field Tt, the label Dj of the datum to be analyzed and the value "Val" of this datum extracted from the memory of the terminal. These blocks are stored one after another in an area of the memory of the terminal called "ZDR." The content of this area is sent to the network at the end

of the transaction or when there is a request by the network for self-diagnostic data. Once all the data have been transmitted, the area ZDR is cleared so that it can be reused when a there is new insertion of a card.

If Sk is equal to "2" (step 13), the datum Dj must be sent to the printer of the terminal for printing. In this case, a message is created in the software buffer of the printer; it is composed of a text (ASCII code) indicating the nature of the datum, for example "AMOUNT," followed by the decimal or hexadecimal value of the datum Dj; the message ends with a separator and a "CARRIAGE RETURN - SKIP LINE." It is possible to gather all of the self-diagnostic messages and print them at the end of the transaction.

If Sk is equal to "3" (step 14), the datum must be sent to the display unit of the terminal for display. In this case, a message is created in the buffer of the display unit, composed of a text (ASCII code) indicating the nature of the datum, for example "AMOUNT," followed by the decimal or hexadecimal value of the datum Dj. The messages corresponding to each element (Tt, Dj, "3") are successively displayed for a certain time set by the program. It is possible to gather all of the messages and display them at the end of the transaction; the scrolling of the messages can be controlled by pressing a key on the keyboard of the terminal.

Once the datum Dj has been processed, the program verifies in step 15, whether there are other triplets in ZTD for which Ti = Tt. If yes, the program loops back to step 11 and processes a new triplet. For each elementary task, the search for triplets is performed by scanning the entire area ZTD. If there are no more triplets (Ti=Tt, Dj, Sk) to be processed, the program, in step 16, continues in sequence and may, in a variant, proceed to another task without executing the steps 17 and 18 described below. If there are no other tasks to be executed, the program loops back to step 3, to wait for a command or for a new insertion of a card.

It is possible to associate a counter initialized with a certain number with the indicator Ind-DT in the terminal so that the self-diagnostic function is only activated for this number of general-purpose card insertions. In order to do this, the operator has pre-entered this number into a specific location (21, Fig. 2) of the programmable memory of the test card, for example, next to the locations (230, 231) of ADD_ZD and ADF_ZD. In this case, this number is stored in the memory of the terminal after the insertion of the test card in step 6. Then this number is decremented (step 17) at the end of each execution of a self-diagnostic function (a YES output from step 16). When it reaches "0," the indicator Ind_DT is set to the inactive position (step 18) and the content of the area ZTD is possibly erased.

If the counter is not installed, steps 17 and 18 do not exist and the self-diagnostic function is executed only once or indefinitely until a new insertion of the test card switches the indicator Ind_DT to the inactive position.

It is possible to avoid the utilization of a test card and to use only general-purpose cards, on condition that they support the special self-diagnostic functions. For this reason, the programmable memory of the general-purpose card contains, in addition to the system area ZS and the user area ZU, an area ZD which is labelled by its start address "ADD_ZD" and its end address "ADF-ZD" (see Fig. 4). The programmable memory of the general-purpose card also contains in its system area, in a location (232), an indicator "Ind_D" which indicates whether or not the self-diagnostic function is active. All of these data ADD-ZD, ADF-ZD, Ind-D are stored in locations (230, 231, 232) of the part ZS of the programmable memory allocated to the operating system. The two address values are determined and written into the area ZD during the customization of the card; this method is simple to implement but has the drawback of requiring the reservation of a sizeable location in all cards that can be used for the self-diagnosis.

Advantageously, the location in the area ZD can be allocated

dynamically by the operating system of the card after the correct entry of the code KD. The operator indicates to the card the number of triplets (Ti, Dj, Sk) or the number of octets to be reserved for ZD. The operating system of the card then searches in the programmable memory for a blank location of sufficient size. If the memory does not contain any such blank location, the operating system returns an error message and interrupts the procedure for entering the self-diagnostic data. In the opposite case where there is sufficient space, the operating system stores the start address "ADD_ZD" and the end address "ADF-ZD." It will be seen below how, after the execution of the self-diagnostic function, it is possible to erase the presence of the area ZD, thus releasing this memory space.

The same is true for the test card. A security procedure is provided in order to prevent a defrauder from being able to use a general-purpose card to enter self-diagnostic data. A mechanism of the challenge-response type with an algorithm and a secret code makes it possible to authenticate the operator and authorize the writing and reading (it will be seen why below) of the triplets in ZD.

In the case where a general-purpose card is used to transmit the self-diagnostic data, the code Sk can assume a fourth value 4; this value indicates that the value of the datum Dj to be verified has been written into the card. In this case, a fourth field located at an address "Adr-V" is allocated at the end of the triplet (Ti, Dj, Sk=4), and thus quadruplets are stored. The size of this field corresponds to that of the data to be written; the operator must therefore specify the number of octets "Nb-V" in this fourth field, and its content is initially constituted by the write address (Adr-V), then after the output by the value "Val," as seen below. The fifth triplet (225) of Fig. 4 has this structure. When all of the triplets (Ti, Dj, Sk) (220 through 225) have been entered into the area ZD, the indicator "Ind_D" is set to the active position, thus indicating that the self-diagnostic function is active in this card.

Fig. 5 shows the sequence of operations when the card described above is inserted into a terminal. Step 1 is the initialization of the terminal after it is powered up and step 2 is the phase for waiting for the insertion of a card; the program continues when the card is recognized as being compatible with the application through the terminal's recognition of the presence of the necessary information. In this step 2, the program performs the selection of the entity corresponding to the application. Unlike the test card, when the general-purpose card is inserted by the bearer, the latter can be completely unaware that the self-diagnostic function is active.

In step 3, the terminal tests whether the indicator Ind_D in the card is set to the active position and thus whether the self-diagnostic function is operational. The indicator can be sent either by a particular value in the octets transmitted by the card during the power-up phase, or by a particular value transmitted during the selection of the entity corresponding to the application used in the card. If Ind_D is active, the program proceeds to step 4. During this step 4, the area ZD is read with the aid of the two address values ADD-ZD and ADF_ZD and all of the triplets read in the card are stored in the memory ZTD of the terminal. If the triplet comprises a datum Sk whose value is "4," the operating system of the card returns, in addition to the three values Ti, Dj and Sk, the address "Adr-v" of the fourth field reserved for writing the datum into ZD and the number of octets "Nb_v" in this field. For security reasons, read access to the area ZD of the card is only granted by the operating system of the card if the indicator Ind_D of the card is active. Once all of the information contained in the area ZD has been stored in the memory area ZTD of the terminal, the terminal sets its self-diagnostic indicator Ind-D to the active position (step 5). Steps 3, 4 and 5 are parts of the sequence for initializing the dialogue between the general-purpose card and the terminal and are executed before the execution of the application program.

As stated above, the application software is broken down

into elementary tasks which can be tested individually. At the end of the execution of each task, which can be labelled by a number Tt (step 6) for example the basic software resumes control and tests whether the diagnostic indicator inside the terminal is active (step 7). If it is active, in step 8 the program searches to see if there is an element (Ti, Dj, Sk) stored in the area ZTD that has a field value Ti equal to that of Tt. If yes, there is a datum (labelled Dj) to be verified as a result of the task that has just been executed; the value of this datum is then temporarily stored in the memory of the terminal. As a function of the value of Sk, it is processed in the following way (step 9):

If Sk is equal to "1" (step 10), the datum Dj must be sent by the terminal to the network. A block of three data is then constituted by the value of the field Tt, the label Dj of the datum to be analyzed and the value "Val" of this datum extracted from the memory of the terminal. These blocks are stored one after another in an area of the memory of the terminal called "ZDR." The content of this area is sent to the network at the end of the transaction or when there is a request from the network for the self-diagnostic data. Once all of the data has been transmitted, the area ZDR is cleared so that it can be reused when a there is a new insertion of a card.

If Sk is equal to "2," the datum must be sent to the printer of the terminal, and the program continues with step 11. During this step 11 a message is created in the buffer of the printer, composed of a text (ASCII code) indicating the nature of the datum, for example "AMOUNT," followed by the decimal or hexadecimal value of the datum Dj, and the message ends with a separator and a "CARRIAGE RETURN - SKIP LINE." Advantageously, it is possible to gather all of the messages and print them at the end of the transaction.

If Sk is equal to "3," the datum Dj must be sent to the display unit of the terminal, and the program continues with step 12. During this step 12, a message is created in the buffer of the

display unit, composed of a text (ASCII code) indicating the nature of the datum, for example "AMOUNT," followed by the decimal or hexadecimal value of the datum Dj. The messages corresponding to each element (Tt, Dj, "3") are successively displayed for a certain time set by the self-diagnostic software. Advantageously, it is possible to gather all of the messages and display them at the end of the transaction; the scrolling of the messages can be controlled by pressing a key on the keyboard of the terminal.

If Sk is equal to "4," the datum must be stored in the card, and the program continues with step 13. During this step 13, a fourth field is reserved for this purpose in ZD, the address "Adr-v" of this fourth field and the number of octets "Nb_v" in this field having been stored in the area ZTD during the loading of the self-diagnostic data into the terminal. The terminal then sends the card a write command with the following parameters:

Write address: Adr-v

Number of octets to be written: Nb-v

Value to be written: "Val" of the datum Dj.

The write operation in the area ZD is only authorized by the operating system of the card when the fourth field of a triplet of the Sk=4 type corresponds to the Ti. In the case where the triplet Ti of the card is not of the Sk=4 type, the write operation is denied. An execution report is systematically returned to the terminal after each write command; if the latter has not been successful, the terminal warns the user with a message. The utilization of the stored data is explained below. A variant is comprised of temporarily storing all the values of the data Dj of the Sk=4 type and of executing the commands for writing the values at the end of the transaction.

Once the datum Dj has been processed, in step 14, the program verifies whether there are other triplets in the area ZTD for which Tt=Ti. If yes, the program loops back to step 9 and processes a new triplet. For each elementary task, the search for triplets is performed by scanning the entire area ZTD. If there

are no more triplets (Ti-Tt, Dj, Sk) to be processed, the program proceeds to step 15, searching for another task to be executed. If there are no other tasks to be executed, the program loops back to step 2 to wait for a new insertion of a card.

In the case where a general-purpose card is used to transmit the self-diagnostic data, the self-diagnostic function should be able to be executed only once. In effect, the operator may intend to perform only one test of the card reader terminal, after which the data must not leave the terminal. Moreover, if the datum to be verified has a field Sk of the 4 type, a single storage is not possible. In order to start the function another time, the card must be reprogrammed by the operator. To prevent it from being used several times with the self-diagnostic function, just after the reading of the area ZD by the terminal, the operating system of the card will set the indicator Ind_D to the inactive state. For security reasons, it can also erase all the triplets of the Sk=1, 2 and 3 types. The indicator Ind-D is inactive, and the reading of the area ZD is no longer possible.

The data corresponding to the type Sk=4 are processed when the general-purpose card enters a terminal authorized to read it, that is, a terminal that is authenticated in the same way as when the self-diagnostic data are written. Once all the triplets have been read, the total erasure of the area ZD can be carried out. This erasure can be triggered by a special command or during the first write operation in the area ZD. The erasure, which is justified for security reasons, makes it possible to release the space occupied by the area ZD. This location can be used by the application.

This "one-shot" style of operation of the self-diagnostic function can be advantageous when a person reports his credit card to a bank branch, declaring that his card "does not work" in a certain type of payment terminal. The branch will store the self-diagnostic data in this card, specifying that the data of the transaction (amount, date, certificate value) are to be written into the card by setting the third field Sk, in each

triplet corresponding to a task to be recorded, at the value "4" (Sk=4). The person returns to the merchant where the terminal to be tested is located, executes a transaction and returns to his branch, which analyzes the information stored in ZD, or has it analyzed remotely.

One-shot operation is also advantageous for verifying the operation of a terminal suspected of fraud. A banking institution discovers that transactions in a terminal have been credited without there having been any request to debit client accounts. The banking institution sends inspectors provided with general-purpose cards with the self-diagnostic function. Upon their return, the data loaded into the card are analyzed.

Another example: a banking institution may find it advantageous to quickly learn the time and the place where the card is used for the first time. For this reason, before the card is delivered to its bearer, it contains two triplets of the Sk=1 type, in which each third field Sk is at the value "1." In the area ZD, with the datum Dj corresponding to the date of the transaction and the datum Dj corresponding to the identity of the terminal. During the first transaction, the two blocks (Ti, Dj, "date") and (Ti, Dj', "terminal ID") will be sent to the network immediately.

CLAIMS

1. A terminal equipped with an application program, with at least one output constituted either by a display, or by a printer, or by a communications network, or by a portable object, and comprising a portable object reader, and cooperating through this reader with a portable object equipped with a non-volatile memory area (ZD) containing data, characterized in that it comprises means for reading or storing, in its memory, self-diagnostic or supervisory data (Ti, Dj, Sk) and means for sending said data to outputs (1-4) specified as a function of information supplied by the self-diagnostic or supervisory data following the execution of at least one task Tt of its application program in connection with the portable object.

2. The terminal according to claim 1, characterized in that the means for sending the self-diagnostic data are activated a certain number of times.

3. The terminal according to claim 1, characterized in that the means for sending the self-diagnostic or supervisory data comprise means for writing into the portable object connected to the terminal.

4. The terminal according to claim 1, characterized in that the self-diagnostic or supervisory data are constituted by at least one triplet (Ti, Dj, Sk) of information corresponding, for a first piece of information (Ti) to a predetermined task of the application program, for the second piece of information (Dj) to a data type correlated to the task executed and to be presented to an output, and for the third piece of information to a value (Sk) for specifying the output to which the data type must be presented among those present in the terminal.

5. The terminal according to claim 1, characterized in

that it has a means for testing for the presence of self-diagnostic or supervisory data in a portable object and for initiation of the reading and the storage of these data in a specific area ZTD of the memory of the terminal.

6. The terminal according to claim 1, characterized in that it comprises means for entering self-diagnostic or supervisory data into a portable object.

7. A process for supervising the operation of a terminal or for the self-diagnosis of a terminal from a triplet of information corresponding, for a first piece of information, to a predetermined task of an application program executed either by a portable object or by a terminal, for the second piece of information to a data type correlated to the task executed and to be presented to an output, and for the third piece of information to a value for specifying the output among those present in the terminal, characterized in that it is comprised of:

- executing (9) a task of the application program in the terminal;
- testing (9) an indicator either in the terminal or in the portable object to determine whether a self-diagnostic or supervisory function is operational, then in the case of a positive response;
- searching (9) in the memory of either the portable object or the terminal to see if among the triplets of information stored there is a triplet wherein the first piece of information corresponds to the predetermined task executed by the terminal or the card;
- sending to the output specified by the triplet thus read the value of the datum correlated to the task executed and to be labelled by the second piece of information of the triplet.

8. The process according to claim 7, characterized in that it comprises a testing step (16) comprised of determining whether

there are other tasks to be executed, and following the execution of these tasks, of searching for all the triplets of information corresponding to the execution of this task.

9. The process according to claim 7, characterized in that it comprises a step for reading from a portable object storing in its non-volatile memory a plurality of triplets and a step (6) for storing these triplets in a non-volatile memory area of the terminal, followed by a step (7) for activating an indicator of an active self-diagnostic or supervisory function.

10. The process according to claim 7, characterized in that it comprises a testing step (3, 4) for determining whether the portable object is a card specific to the self-diagnostic or supervisory function or a so-called general-purpose card.

11. The process according to claim 7, characterized in that the self-diagnostic or supervisory data are constituted by a fourth field of information containing in the portable object initially the write address (Adr-V), the number of octets to be written (Nb-V), and after the self-diagnosis operation, the value to be written (Val).

12. A portable object that can be used with a terminal equipped with an application program, with at least one output constituted either by a display, or by a printer, or by a communications network, or by a portable object, means for reading or storing, in its memory, self-diagnostic or supervisory data (Ti, Dj, Sk) and means for sending said data to outputs (1-4) specified as a function of information supplied by the self-diagnostic or supervisory data following the execution of at least one task Tt of its application program in connection with the portable object, characterized in that it comprises a microprocessor card operating by means of an operating system stored in the card and comprising a non-volatile memory

containing at least one triplet (Ti, Dj, Sk) of information in a predetermined area ZD of this non-volatile memory, whose location is defined by address fields located in the memory part used to store the operating system.

13. The portable object according to claim 12, characterized in that the part (25) of non-volatile memory used to store the operating system also comprises, in a memory field, a piece of information constituting a counter (21) of utilizations of the self-diagnostic function.

14. The portable object according to claim 12, characterized in that the memory area (25) storing the operating system comprises a field which makes it possible to store an indicator (232) of the activation of the self-diagnostic or supervisory function.

		Value of the bearer's code D1 Valeur du code porteur	Identity of the bearer D2 Identité du porteur	Value of the transaction D3 Valeur de la transaction	Date of the transaction D4 Date de la transaction
Verification of card authorization T1	Vérification habilitation carte	SK	X SK=1		
Authentication of the bearer T2	Authentification du porteur	X SK=2			
Aquisition of the transaction data T3	Acquisition des données transaction			X SK=1	
Validation of the transaction T4	Validation de la transaction			X SK=3	X SK=1

Fig.1

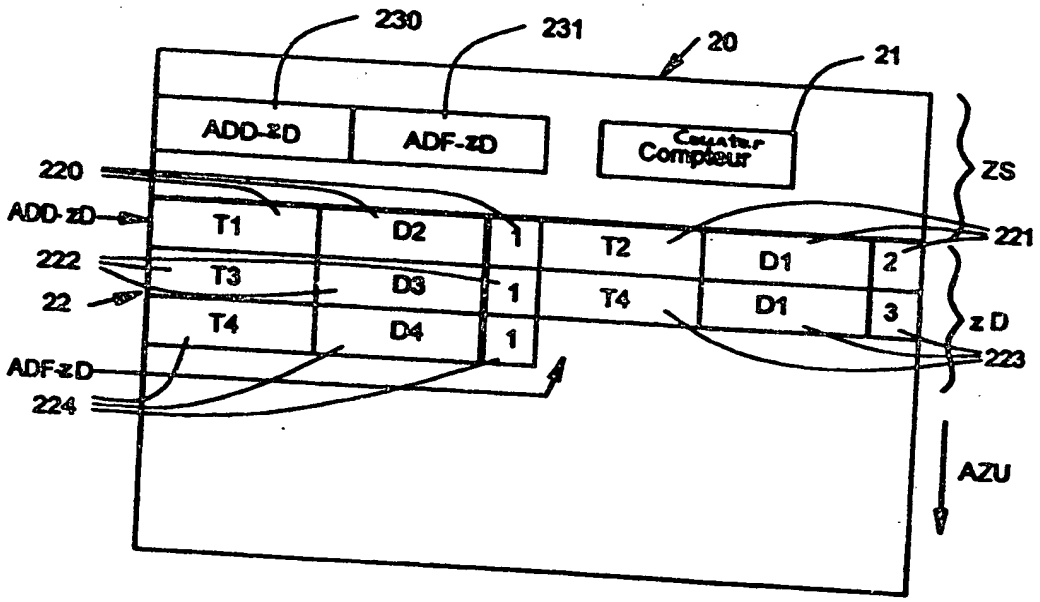
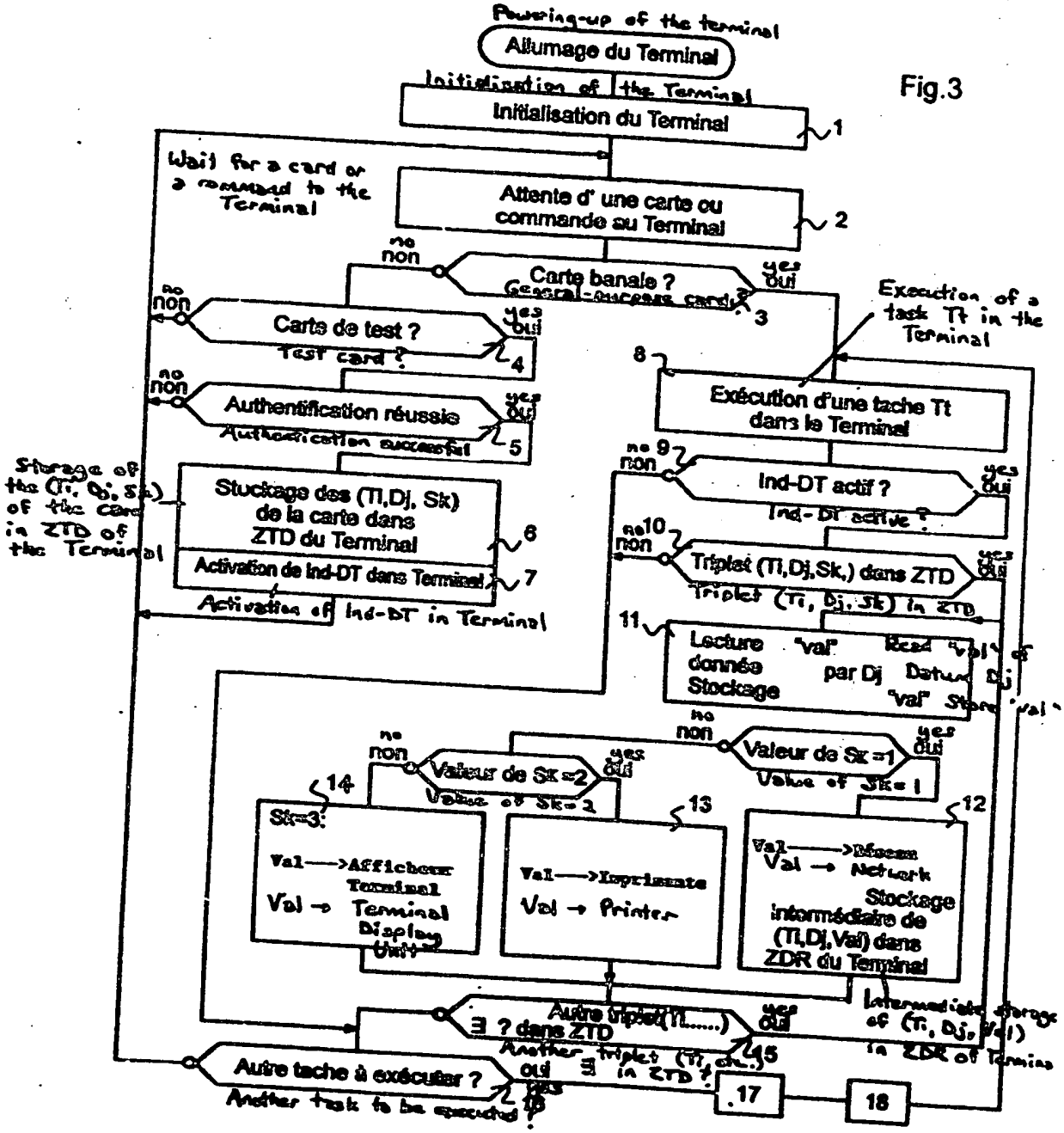


Fig.2

Fig.3


Powering-up of the terminal
Allumage du Terminal
Initialisation of the Terminal
Initialisation du Terminal
1
Wait for a card or a command to the Terminal
Attente d' une carte ou commande au Terminal
2
non
Carte banale ?
General-purpose card ?
oui
3
non
Carte de test ?
Test card ?
oui
4
non
Authentification réussie
Authentication successful
oui
5
Storage of the (Ti, Dj, Sk) of the card in ZTD of the Terminal
Stockage des (TI,Dj, Sk) de la carte dans ZTD du Terminal
6
Activation de Ind-DT dans Terminal
7
Activation of Ind-DT in Terminal
Execution of a task Tt in the Terminal
8
Exécution d'une tache Tt dans le Terminal
non
9
Ind-DT actif ?
Ind-DT active ?
oui
non
10
Triplet (TI,Dj,Sk,) dans ZTD
Triplet (Ti, Dj, Sk) in ZTD
oui
11
Lecture "val" donnée Stockage
Read "val" Datum Dj "val" Store
non
Valeur de Sk=1
Value of Sk=1
oui
non
Valeur de Sk=2
Value of Sk=2
oui
14
Sk=3:
Val ——> Affichage Terminal
Val → Terminal Display Unit
13
Val ——> Imprimante
Val → Printer
12
Val ——> Réseau
Val → Network
Stockage intermédiaire de (TI,DJ,Val) dans ZDR du Terminal
Intermediate storage of (Ti, Dj, Val) in ZDR of Terminal
Autre triplet(TI,.....) ∃ ? dans ZTD
Another triplet (Ti, ...) ∃ in ZTD ?
non
oui
15
Autre tache à exécuter ?
Another task to be executed ?
oui
16
non
17
18

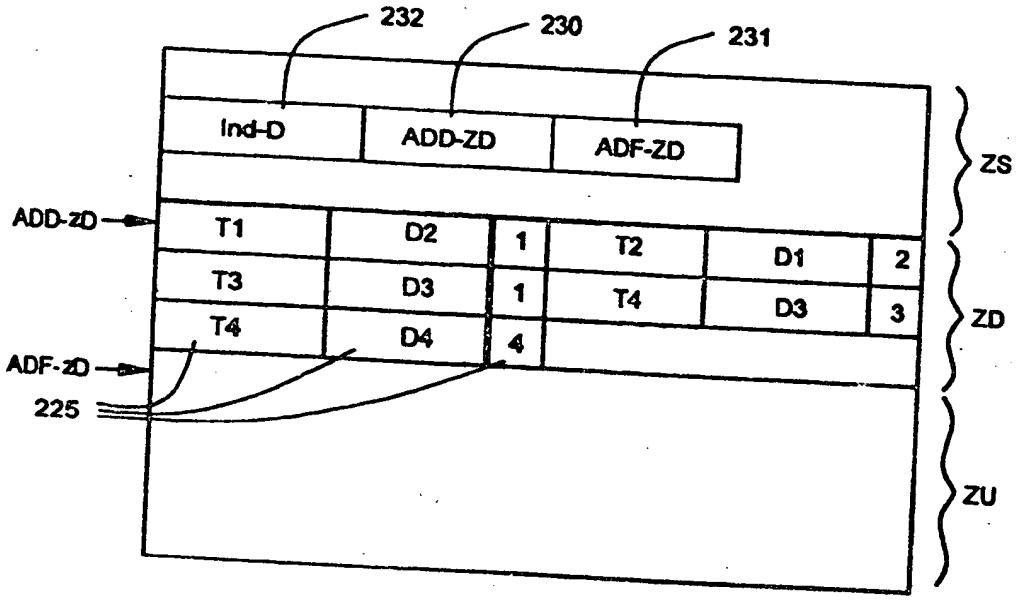
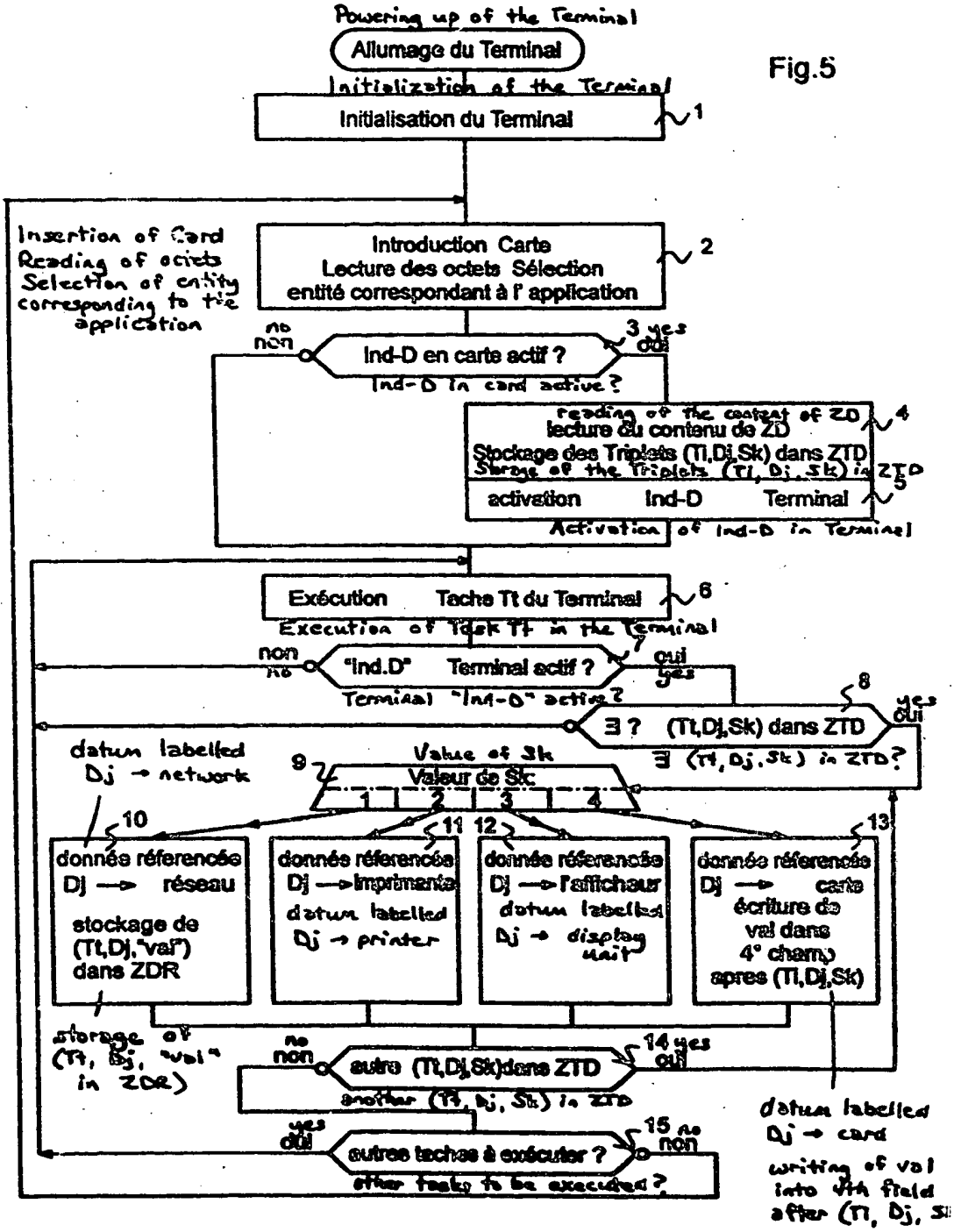

232
230
231
Ind-D
ADD-ZD
ADF-ZD
ZS
ADD-zD
T1
D2
1
T2
D1
2
T3
D3
1
T4
D3
3
ZD
T4
D4
4
ADF-zD
225
ZU

Fig.4

Fig.5


Powering up of the Terminal
Allumage du Terminal
Initialization of the Terminal
Initialisation du Terminal
1
Insertion of Card
Reading of octets
Selection of entity corresponding to the application
Introduction Carte
Lecture des octets Sélection
entité correspondant à l' application
2
non
Ind-D en carte actif ?
Ind-D in card active?
3 yes oui
reading of the content of ZD
lecture du contenu de ZD
4
Stockage des Triplets (Tl,Dj,Sk) dans ZTD
Storage of the Triplets (Tl, Dj, Sk) in ZTD
activation Ind-D Terminal
5
Activation of Ind-D in Terminal
Exécution Tache Tt du Terminal
6
Execution of Task Tt in the Terminal
non
"Ind.D" Terminal actif ?
Terminal "Ind-D" active?
7
oui yes
8
∃ ? (Tl,Dj,Sk) dans ZTD
∃ (Tl, Dj, Sk) in ZTD?
datum labelled Dj → network
Value of Sk
Valeur de Sk
9
1 2 3 4
10
donnée référencée Dj → réseau
stockage de (Tl,Dj,"val") dans ZDR
11
donnée référencée Dj → Imprimante
datum labelled Dj → printer
12
donnée référencée Dj → l'afficheur
datum labelled Dj → display unit
13
donnée référencée Dj → carte
écriture de val dans 4° champ apres (Tl,Dj,Sk)
storage of (Tl, Dj, "val") in ZDR
datum labelled Dj → card
writing of val into 4th field after (Tl, Dj, Sk
no non
autre (Tl,Dj,Sk) dans ZTD
another (Tl, Dj, Sk) in ZTD
14 yes oui
yes oui
autres taches à exécuter ?
other tasks to be executed?
15 no non